48K SPECTRUM

ADVENTURE BUILDER SYSTEM

HELP-LINE: 0674-74239

TARTAN SOFTWARE
61, Ballie Norrie Crescent
MONTROSE
Angus
Scotland
DD10 9DT

48K SPECTRUM ADVENTURE BUILDER SYSTEM

An adventure written entirely in BASIC has response times which compare very badly with machine coded programs. The "slow" sections are readily identifiable as:- the parsing routine, word recognition through long FOR... NEXT loops, through many IF....THEN lines and identifying which few objects should appear at a location or in the player's INVENTORY.

The ADVENTURE BUILDER SYSTEM (ABS) can substantially decrease the above delays such that an adventure written predominantly in BASIC can operate at a speed approaching that of 100% machine-coded programs.

The ABS consists of two programs.....one to generate the required CODE with the other being a short BASIC core program which you will expand and which will utilise this CODE.

Before attempting to use the system, it is advisable for the adventure to be outlined to a reasonably advanced stage. The map should be constructed with each location numbered and all objects well identified as to description and starting location. Details on how to use the system will make reference to the short demonstration adventure for which the DATA and map are provided in this booklet.

The final BASIC adventure program follows the convention of:-

a) PRINTs location description
b) PRINTs any objects visible at this location
c) Asks "What next?" and awaits input
d) Accepts input from keyboard and PRINTs it on screen
e) Analyses input (usually into verb and noun)
f) Identifies the allocated number of that verb and noun
g) Sends control of the program to a sub-routine specific to that verb

The CODE GENERATOR program already includes DATA for movement input (GO NORTH [or N] etc) and also for the following commonly used verbs:-

GET (or TAKE)
DROP
LOOK (or L or R)
EXAMINE
QUIT (or STOP)
INVENTORY (or I or LIST)
SAVE
LOAD

You may wish to add further synonyms (eg INVE for INVENTORY).

METHOD OF OPERATION

1. Prepare (by GOTO 40 *), a tape with the following loader program:-*

```
10 CLEAR 53000
20 LOAD ""CODE
30 LOAD ""
40 SAVE "loader"LINE 10
```

VERIFY and do NOT rewind the tape.

2. LOAD *the Blank CODE GENERATOR program, add your adventure DATA (how to do this will be explained later) then* SAVE *on to the above tape by the direct command* GOTO 9990.....VERIFY, *rewind and reset computer. This is TAPE A. If necessary the DATA may be modified at a later date, but due cognisance MUST be given to any alteration in the CLEAR number when the DATA is generated.*

3. LOAD *the* BASIC *core program and copy it on to a blank tape by* GOTO 9990, VERIFY, *do not rewind the tape. Reset computer. This is TAPE B.*

4. LOAD *TAPE A and supply the information required by the on screen prompts then* SAVE *the generated DATA on TAPE B when instructed. Make a careful note of the CLEAR number (xxxxx). Rewind TAPE B and reset computer.*

5. Prepare another blank tape (by GOTO 30 *) with the following loader program:-*

```
10 CLEAR xxxxx
20 LOAD ""
30 SAVE "loader"LINE 10
```

VERIFY and do not rewind the tape. Reset computer. This is the FINAL tape!

6. Enter as a direct command: CLEAR xxxxx:LOAD "" *and* LOAD *TAPE B. STOP the tape when prompted, delete LINE 9949 then enter as a direct command:* GOTO 9950 *and restart the tape.*

7. BREAK *and alter line* 9991 *to:-*

```
9991 SAVE "SYSTEM"CODE save,len
```

8. Add the remainder of the BASIC program (verb sub-routines, screen presentation details etc).

9. SAVE *the adventure on to the FINAL tape by* GOTO 9990. *This* SAVE *can be performed with only part of the BASIC program added, then* LOADed *and re-*SAVEd *at a later date for the remainder.*

In order to use the ABS, the DATA for the adventure must be added to the CODE GENERATOR program (always use a COPY of the program NEVER the original), and the on-screen prompts satisfied in the following manner. Responses for the DATA for the Demonstration Adventure are indicated after each section heading.

A) HOW MANY VERBS (17)

The DATA for these is entered in lines 1003 to 1999 in the format:-

```
1003 DATA "OPEN",9,"UNLOCK",10
```

As the verb numbers (allocated the variable vb) 0 to 8 are already assigned the first additional verb takes the value 9.

Synonymous verbs (eg SHUT and CLOSE) are allocated the SAME number but count as separate verbs when responding to HOW MANY VERBS?

B) VERB LENGTH CHECK (5)

The ABS offers a choice (between 3 and 7) for the number of leading letters which will be used to determine a match between input and database. The selection of 4 or 5 is usual. If 3 were selected then the program would be unable to distinguish between verbs such as SHOW and SHOOT.

C) HOW MANY OBJECTS (9)

This refers to the TOTAL number of objects which could be included in the INVENTORY of the player (not necessarily all at the same time).

For DATA entry details, see section F).

D) NOUN LENGTH CHECK (5)

This acts in the same way as the VERB LENGTH CHECK.....respond with a number between 3 and 7. The value can be different from that chosen for VERBS.

E) MAXIMUM LENGTH OF OBJECT DESCRIPTION (22)

Respond here with the number of characters in the longest object description.

F) HOW MANY OBJECTS WHICH CAN HAVE AN ALTERED DESCRIPTION (2)

This refers to items such as "A small torch" and "A lit torch". The DATA for this type of object is entered in lines 2000 to 2018 in the following manner:-

```
2000 DATA "COAT",1,"A tartan COAT","A tartan COAT (worn)"
2001 DATA "TORCH",2,"A small TORCH","A lit TORCH"
```

is The word which would be typed by the player, the assigned number, the original description, the changed description.

Within the CODE GENERATOR the description of objects is temporarily held in the string array o$(x,y), where x is the total number of objects and y is the length of the longest object description.

It is ABSOLUTELY ESSENTIAL that the objects which can alter during the course of an adventure use the FIRST and LAST numbers of this string array. This is simply accomplished by placing these objects in the appropriate position in your list of objects. See list of DATA for the Demonstration Adventure where "A tartan COAT" and "A small TORCH" are objects 1 and 2 while the corresponding changed versions are objects 8 and 9.

The DATA for the remaining objects [from Section C)] are entered in lines 2020 to 2188 in the format:-

```
2020 DATA "BOX",3,"A wooden BOX","SPADE",4,"A short SPADE"
```

G) OTHER WORDS (13)

The first assigned number for these words (DOOR, STAIRS etc) is calculated as follows:-

TOTAL number of objects + 7 (ie 9 + 7 = 16 for Demonstration Adventure)

The DATA is entered in lines 2190 to 2988 in the format:-

```
2190 DATA "DOOR",16,"CUPBOARD",17
```

H) MAXIMUM INVENTORY (3)

Respond here with the maximum number of items which the player is allowed to carry at one time.

I) NUMBER OF LOCATIONS (15)

Respond with the total number of locations in the adventure.

J) SCROLL ROUTINE (B)

The ABS contains a split-screen technique which can be utilised in two ways:-

1) If it is required that the top X lines remain on screen, then an entry here of "C" , followed by X, will "scroll behind" these lines when the screen is full.

2) However if the number of lines required to remain on the screen is variable (ie after the HERE YOU CAN SEE *section), then this is accomplished by the selection of "B".*

Selection of "A" will provide the normal SPECTRUM full-screen scroll.

K) STARTING LOCATION (7)

Respond with the number of the location in which the adventure commences.

L) COLOUR OF BACKGROUND (0)

Respond with the colour value (0 to 7) for the main screen.

M) CENTRALISATION OF PRINT (1)

The printing of HERE YOU CAN SEE *and* YOU HAVE WITH YOU *followed by the relevant list of objects can be located in the centre of the screen or at the extreme left of the screen.....respond with either 0 or 1 (as appropriate) when prompted.*

N) BRIGHT VALUE (1)

Respond with the value of BRIGHT (0 or 1) for the main screen.

O) RESPONSES from within the Machine Code SYSTEM (1 TERMINAL and 6 ROUTINE)

The DATA for these (in numerical order) is entered in line 7090 and onwards as follows:-

```
7090 DATA 6,"You really need som
e light in   here!
```

The number refers to the INK colour for this message and the text is arranged for tidy printing on screen.

Three types of response are considered.....

STANDARD.....in lines 7090 to 7098
TERMINAL.....in line 7100 and onwards
ROUTINE.....in line 7150 and onwards

BUT THEY ARE NUMBERED IN SEQUENCE WITHOUT REGARD TO TYPE

Hence in the DATA for the Demonstration Adventure, as there are 9 STANDARD responses, 1 TERMINAL response and 6 ROUTINE responses, the reference number for the first ROUTINE response is 11 and for the sole TERMINAL response it is 10.
Any response which is identified as a TERMINAL response will automatically be followed on screen by the "Do you want to try again?" phrase.

P) MOVEMENT DATA

The DATA to control movement is entered in lines 3000 to 3998 as shown below. A line such as this example, for location number 7, must be entered for each location.

3007 DATA 7,3,13,8,6,0,0

The first number after DATA is the location number, followed by the numbers of the six locations which would be reached by going North, South, East, West, Up, Down respectively AND IN THAT ORDER. An entry of 0 indicates no exit in that direction.

The DATA entered here takes no account of the need to remove obstacles or unlock doors etc, as such restrictions can be accommodated by the BASIC program, in lines 1005 to 1089. Examples of this are to be found in lines 1005 and 1010 in the Demonstration Adventure.

Q) OBJECT STATUS TABLE

The status of each object will at any one time be described by one of the following values.

a) 0.....the object is "invisible"
b) 99.....the object is in the INVENTORY
c) location number.....the object will be found at this location

The DATA for initial positions of all objects is entered in lines 4000 and onwards in the format:-

4000 DATA 4,10,0,1 etc

This indicates that object 1 is at location 4, object 2 is at 10 and 3 is "invisible", ie it is not in the adventure YET.

These initial values will be altered by your BASIC program as the adventure progresses. How this is achieved is explained later.

R) SAVE AND LOAD ROUTINES

These routines can be used on a temporary basis effecting a SAVE into and a LOAD from RAM or as a permanent measure as usual to tape.

S) LOCATION DESCRIPTIONS

This DATA is entered in line 5001 and onwards as:-

```
5001 DATA 6,"You are in a very t
idy BARN. Thefarmer who runs thi
s farm is    obviously very prou
d of his work          Or is he?"
```

This is how it will appear on the Spectrum screen. The text within the quotes must be carefully constructed with regards to spaces and punctuation so that when PRINTed on the screen during the adventure it does not produce

an untidy appearance. The number after DATA refers to the INK colour for this text. The number may be kept constant for all locations, but for special emphasis a change of presentation can be quite effective.

T) HERE YOU CAN SEE and INVENTORY

The phrases associated with these are to be found in lines 6040 and 6284 for the former and in lines 6080 and 6254 for the latter, currently as follows, (with the numbers referring to the PAPER, INK and BRIGHT values for each phrase):-

```
6040 DATA 5,0,1," Here you can see:-    "
6284 DATA 5,0,1," Nothing of interest   "
```

ALL of the objects require values for PAPER, INK and BRIGHT and these are entered (in order) in line 6140. A set of values MUST be included for each object.
If the choice of PAPER is not the same as the normal screen colour, then pad out the standard phrases with spaces to match the length of the longest object description. This will effect a tidy presentation on the screen.

##########

DEFINED VARIABLES

In order to maximise the use of memory the following variables have been used in place of numbers in the CODE GENERATOR program. Further use of them may help in obtaining the maximum benefit from the program.

0=n	1=j	2=a	3=w	5=ab	6=ac	7=ad	8=b	9=ae
10=af	11=ag	12=ah	13=ai	14=aj	15=ak	16=c	17=al	18=am
19=an	20=ao	22=ap	23=aq	24=ar	31=as	32=d	33=e	35=f
40=g	42=at	48=au	50=av	54=h	58=i	60=k	61=l	62=m
70=ax	79=ay	92=az	99=bj					

100=ba	115=bb	126=bc	137=bd	184=be	190=bf	200=bg	201=o	202=p
205=q	215=u	250=v	251=bi	254=r	255=s	400=t		

HOW TO USE ABS

The following sections explain how the system operates and how it may be used to write your own adventures.

1) ANALYSER ROUTINE

The ABS allows (automatically) a maximum input of 30 characters as numbers or upper case letters.

The analyser routine examines the input and returns to BASIC with an allocated number for the VERB (vb) and NOUN (no) which the BASIC program can utilise. Any unrecognised word returns a value of 200 for VERB or 201 for NOUN. This information can be used in various ways by the BASIC program but in the Demonstration Adventure, either situation produces the all-purpose cop-out of "You can't do that!".

Possible SINGLE word inputs require some further consideration. The parser routine analyses the input by taking the first word as the VERB and the last word as the NOUN so that DROP THE SPADE or DROP THE SHORT SPADE will produce the same response as DROP SPADE. HOWEVER, the one word input of LOOK gives a VERB and NOUN of LOOK with values of vb=4 and no=201. This would produce a response of "You can't do that!" due to the value of no=201. Such an outcome is prevented by including LOOK in the list of NOUNS and assigning it a value. In the Demonstration Adventure it has been given a default value of 99 as have other one-word (or one-letter) inputs. Alternatively, the NOUN number could be a true assigned number for use by the BASIC program. In the Demonstration Adventure such an approach has been used for DIG where it has been assigned the same NOUN number as HOLE as an input of DIG implies DIG HOLE.

2) NUMBER OF LOCATIONS

If your adventure contains more than 65 locations then line 7665 of the BASIC program should be altered to the value of 7005 + (10 x the number of locations)

3) CONTROL OF THE ADVENTURE

The adventure is controlled in the BASIC program by:-

The variable vb (the VERB number)
The variable no (the NOUN number)
AND
The OBJECT STATUS TABLE
The FLAG STATUS TABLE

All other controls are operated by the generated CODE.

a) The OBJECT STATUS TABLE (OST)

The object status is expressed in the BASIC program as (o+x) where x=the number of the object.

Fifty entries are available in this table and each object has at all times an entry here in the position indicated by the number allocated to that object.

The status (0 or 99 or location number, as outlined earlier) of object number x is found by the expression:

```
PEEK (o+x)
```

and a change in status is effected by:

```
POKE (o+x),y
```

where y indicates the new status.

b) The FLAG STATUS TABLE (FST)

The flags are expressed in the BASIC program as (f+x) where x=the flag number.

One hundred entries are provided by this table with a few being reserved for specific uses:-

FLAG (f+99) is used to control the size of the INVENTORY.
FLAG (f+98) is used by the GRAPHICS AID program.
FLAG (f+0) contains the number of the current location.
The first x FLAGS are reserved for use by the x objects which can change description during the course of an adventure. (See example below).

All other FLAGS are available for use by the BASIC program.
A FLAG may also be used as a counter, but only to a maximum value of 255.

A FLAG status is found by the expression:

```
PEEK (f+x)
```

and a change in status is effected by:

```
POKE (f+x),z
```

where x is the flag number and z is the new status.

As the adventure progresses the BASIC program will be required to up-date the values in OST and FST.

At start-up all flags (except (f+0)) have the value 0.

Example: In the Demonstration adventure, the value of FLAGS (f+1) and (f+2) are altered from 0 to 1 (by POKE (f+1),1 *and* POKE (f+2),1 *) when the COAT is worn and the TORCH is lit respectively. As these are the only objects in this adventure which alter their description then the first available FLAG for general use is (f+3) and this is altered from 0 to 1 when the CUPBOARD is open.*

COMPLETION OF THE BASIC PROGRAM

1) VERB SUB-ROUTINES

Lines 202 and 204 extract the values of vb and no from the machine code routines after the input has been (ENTERED). Assuming both vb and no are recognised then line 235 sends the program to the appropriate verb sub-routine. One hundred lines are available for each sub-routine at line 1000 + (vb x 100).

If guidance is required on how to construct these then consult paragraph 5 of this section.

2) PRINTING OF RESPONSES

As it is most likely that a split screen presentation will be selected, it is necessary to check the need to scroll before each line of a response is printed. To remove the need to include such a check before each PRINT statement the responses may be identified as r$(1), r$(2), r$(3) and r$(4) which are then PRINTed by the PRINT routine at line 50. Each r$() must be no longer than 32 characters.

```
.........:LET r$(1)="OK":GOTO 50
```

It is useful to employ this print routine when testing and de-bugging sections of the adventure. However the printing of responses can be incorporated into the machine code system by replacing the statement.....LET r$( )="xxxxxxxxxxxxxxx":GOTO 50.....with....LET r=x:GOTO 50..... where x=the number of the response held in the machine code system. (See section 0) previously).

Seven responses in the Demonstration Adventure have been incorporated into the machine code system.....those in lines 208, 1092, 1100, 1105, 1110, 1915, 7070 and 7150. The difference in "response time" can be observed by comparing the reaction to OPEN CUPBOARD (in FRONT HALL) with DIG HOLE when in the GARDEN.

3) VARIABLES USED BY THE BASIC PROGRAM

The following variables are used within the BASIC program. The lines defining them can be removed after the first SAVE.

vb, no, f, o, MAX, SAVE, LEN, DMOV, TRMO, PRIN, PRCAR, PRMS, PRHY, SYS, SA, LO, QUIT and r$(4,32)

4) LINE NUMBERS

It is IMPORTANT that lines 100 and 195 are retained in the BASIC program as is (except for alteration of detail in line 100). These lines are on occasions pointed to by the the machine code system and any change could be disastrous.

5) CONSTRUCTION OF VERB SUB-ROUTINES

The method of construction of the VERB sub-routines can be observed by studying lines 1000 to 3300 of the Demonstration Adventure. However, for a detailed examination (by way of an example) consider the input of LIGHT TORCH.

This will return a value of 12 for vb and a value of either 2 or 9 for no depending on whether the TORCH is lit or not (this check will have been made automatically by the system code).
For a vb value of 12, control of the program will be directed to the line 2200 (ie 1000 + (12 x 100)) as calculated in line 235.

Translation into English of this section would be:-

Line 2200:

```
IF PEEK (o+9)=99.....If lit TORCH is being carried
AND no=9.....and NOUN = lit TORCH
THEN LET r$(1)="It's already lit!":GOTO 50..formulate response and PRINT it
```

Line 2205:

```
IF no=2.....If NOUN = a small TORCH,
AND PEEK f<>11.....and location is NOT the CELLAR
AND PEEK f(o+2)=99.....and small TORCH is in INVENTORY
THEN POKE (o+2),0:.....then replace the small TORCH in
POKE (o+9),99:.....INVENTORY with the lit TORCH.
POKE (f+2),1:.....Set the FLAG for torch is lit.
LET r$(1)="OK, TORCH now lit.":GOTO 50.....formulate response and PRINT it
```

Line 2210:

Similar to 2205 but player must be in CELLAR this time, so send program to line 100 to print full description as the player is now carrying a lit TORCH.

Line 2215

Any other situation produces the cop-out response!

NB: An improvement would be to include a positive response for the situation where the player did not have the TORCH in the INVENTORY.

VERBS AND NOUNS INCLUDED IN THE CODE GENERATOR

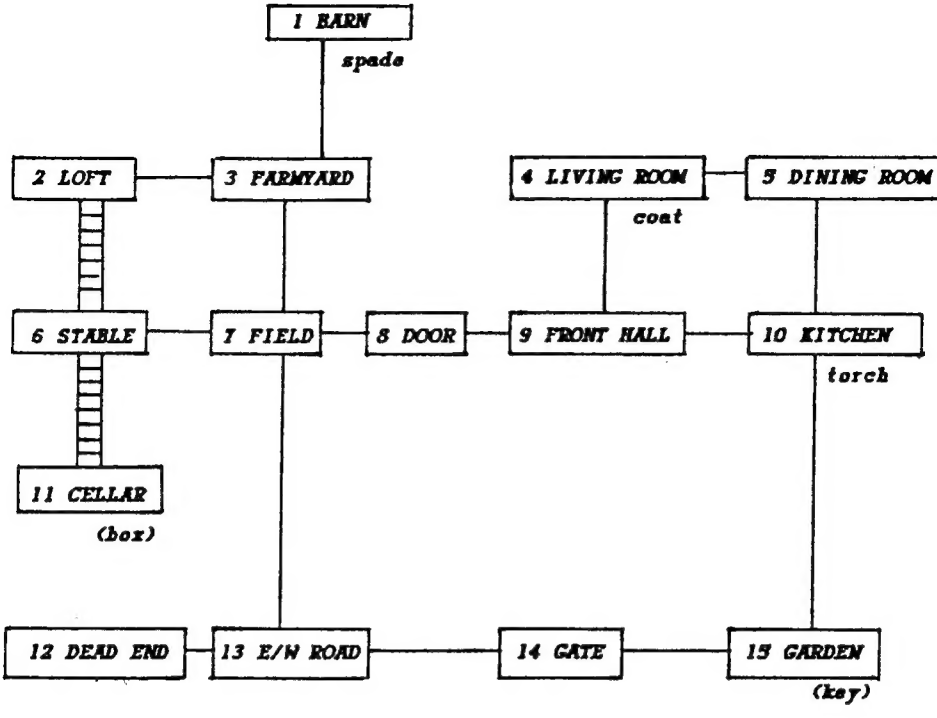
VERB	vb number	NOUN	no number
GO	0	N	1
N	0	NORTH	1
S	0	S	2
E	0	SOUTH	2
W	0	E	3
U	0	EAST	3
D	0	W	4
GET	1	WEST	4
TAKE	1	U	5
DROP	2	UP	5
EXAMINE	3	D	6
LOOK	4	DOWN	6
L	4	LOOK	99
R	4	L	99
INVENTORY	5	R	99
I	5	INVENTORY	99
LIST	5	I	99
QUIT	6	LIST	99
STOP	6	QUIT	99
SAVE	7	STOP	99
LOAD	8	SAVE	99
		LOAD	99

The following verb sub-routines are included in the core BASIC adventure listing

LOOK.....at line 1400
INVENTORY.....at line 1500
QUIT.....at line 1600
SAVE.....at line 1700
LOAD.....at line 1800

The MOVEMENT sub-routine is partially completed only, starting at line 1000. It remains to insert any lines to account for temporary blocks such as locked doors or guarding monsters!

MAP OF THE DEMONSTRATION ADVENTURE



****

OBJECT DATA FOR DEMONSTRATION ADVENTURE

NUMBER	OBJECT DESCRIPTION	RECOGNITION WORD	STARTING STATUS
1	A tartan COAT	COAT	at location 4
2	A small TORCH	TORCH	at location 10
3	A wooden BOX	BOX	0
4	A short SPADE	SPADE	at location 1
5	A bronze KEY	KEY	0
6	Splinters of WOOD	WOOD	0
7	A large DIAMOND	DIAMOND	0
8	A tartan COAT (worn)	COAT	0
9	A lit TORCH	TORCH	0

OTHER WORDS IN DATABASE

NUMBER	WORD	NUMBER	WORD
16	DOOR	23	GARDEN
17	CUPBOARD	24	FIELD
18	LAWN	25	SAFE
19	STAIRS	26	PATCH
20	LADDER	27	JUMP
21	GATE	22	DIG
22	HOLE		

VERBS IN DATABASE *>>>>>>>>>> MOVEMENT DATA <<<<<<<<<<*

vb number	VERB	location	N	S	E	W	U	D
9	OPEN	1	0	3	0	0	0	0
10	UNLOCK	2	0	0	0	0	0	6
11	DIG	3	1	7	0	0	0	0
12	LIGHT	4	0	9	5	0	0	0
13	EXTINGUISH	5	0	10	0	4	0	0
14	CLIMB	6	0	0	7	0	2	11
15	DESCEND	7	3	13	8	6	0	0
16	CLOSE	8	0	0	9	7	0	0
17	TOSS	9	4	0	10	7	0	0
18	FILL	10	5	15	0	9	0	0
19	LOCK	11	0	0	0	0	6	0
20	WEAR	12	0	0	13	0	0	0
21	REMOVE	13	7	0	14	12	0	0
22	JUMP	14	0	0	15	13	0	0
23	USE	15	10	0	0	14	0	0
14	ASCEND							
16	JUMP							

****

FLAG STATUS TABLE.....ASSIGNED VALUES

FLAG NUMBER	CONDITION FOR VALUE = 1
1	COAT worn (fixed allocation)
2	TORCH lit (fixed allocation)
3	CUPBOARD open
4	DOOR open
5	HOLE in field
6	HOLE in garden
7	GATE open
8	SAFE unlocked
9	Found BOX
10	Found KEY
11	First OPEN CUPBOARD input
.	
.	
.	
.	
.	
.	
.	
.	
99	INVENTORY (fixed allocation)

OTHER CONSIDERATIONS

1) VARIABLES

When memory becomes a problem, one of the first considerations is usually to replace frequently used numbers with variables. For example, 0 and 1 will usually abound in a BASIC adventure listing and the use of ..... LET n=0 and LET j=1, followed by the replacement of all the 0's and 1's by n and j respectively will effect a considerable saving of memory.
Such an approach, when applied to an extreme extent can have an adverse affect on response time. It is therefore necessary to strike a considered balance between speed and available memory.

When all the variables have finally been defined, then all lines such as 9960 can be deleted.

2) SUB-ROUTINE STRUCTURE

The length of each sub-routine will influence the response time if the program is required to plough through a vast number of lines before reaching a conclusion. Careful construction can minimise this.
Consider the EXAMINE sub-routine.....this is likely to be one of the longest in terms of lines BUT acceptable response times can be obtained by dividing the sub-routine into several sections. In the first instance the EXAMINE command can be considered with two groups of objects.....those which can be carried (e.g. the KEY) and those which are included in the location description (e.g. the DOOR). A line such as 1300 in the Demonstration Adventure produces two shorter EXAMINE sub-routines, each of which could be further sub-divided if necessary.

3) PERSONALISATION

Certain features incorporated in the ADVENTURE BUILDER SYSTEM can readily be altered to provide characteristics of your own choice.

a) Standard Phrases

The Standard phrases (e.g. HERE YOU CAN SEE) can be altered by changing the appropriate DATA in the CODE GENERATOR. When the chosen PAPER colour is not the same as the main background screen colour then care must be exercised over the use of spaces to ensure that all phrases that are likely to be printed one under another are of the same length.

b) Input Prompt and Beeps

By altering the values in the following locations it is possible to "personalise" your own masterpiece:-

LOCATION	CURRENT VALUE	RESULT
64510	62	Prints >
64528	42	Prints *
64633	50)	
64634	0)	
64636	200)	
64637	1)	Input keyboard BEEP
64049	5)	
64050	1)	
64052	125)	
64053	1)	Wrong-key-press BEEP

4) SCREEN PRESENTATION

An adventure with good story-line, atmosphere and puzzles can be ruined by poor screen presentation, so considerable care should be taken to provide a good impression.

It appears that it has become almost universally accepted that BORDER and PAPER are set to the same colour. When BRIGHT has the value of 1 for the main screen then only BLACK (and to a lesser extent BLUE) provides an acceptable appearance when BORDER and PAPER are the same. The value of BRIGHT is required by ABS in order that the printing performed from within the machine code system can provide an acceptable presentation.